US-PAT-NO: 6125383

DOCUMENT-IDENTIFIER: US 6125383 A
See image for Certificate of Correction

TITLE: Research system using multi-platform object oriented program language for providing objects at runtime for creating and manipulating biological or chemical data

---------- KWIC ---------

Application Filing Date - AD (1):
19970611

Brief Summary Text - BSTX (12):
Furthermore, researchers in industry face significant security issues. Sequence data (that may have cost millions of dollars to collect) cannot be sent over the extremely public **Internet** where anyone might be listening. Consequently, many useful tools for sequence analysis (e.g., those provided over the **Internet** by the National Center for Biotechnology Information (NCBI), such as BLAST or Entrez) may be undesirable to use for researchers in industry due to the lack of security.

Detailed Description Text - DETX (18):
According to an exemplary embodiment of the present invention, Java and CORBA (Common Object Request Broker Architecture) are employed to carry out the present invention. Java is an object-oriented, distributed secure, architecture neutral language. Java provides for object-oriented design which facilitates the clean definition of interfaces and makes it possible to provide reusable "software ICs." As will be mentioned in greater detail below, object-oriented design is a technique that focuses on the data (=objects) and on the interfaces to it. For example, an "object-oriented" biotechnologist would be mostly concerned with the DNA strand he/she is designing, and secondarily with the tools used to make it; a "non-object-oriented" biotechnologist" biotechnologist would think primarily of his tools. Java has an extensive library of routines for copying easily with **TCP/IP** protocols like HTTP and FTP. Java applications can open and access objects across a network via URLs with the same ease to which programmers are accustomed accessing a local file system.

Detailed Description Text - DETX (23):
Thus, the use of Java in the present invention provides for an internal "intranet" that operates exclusively within the walls of a company. Java affords for a server to send bioinformatics and/or chemiinformatics programs over the network as easily as traditional servers send data. These programs can display and manipulate data, such as DNA sequences on a client computer. The present invention through the use of Java affords for multiplatforming.

That is the same bioinformatics and/or chemiinformatics programs can be run on substantially all computers--the same applet (a program designed to be delivered through a browser) can work on a Macintosh, a Windows 95 machine, a Sun workstation, etc. To effect such multiplatforming, a network connection 70 and a network browser (not shown) such as Netscape Navigator or Microsoft **Internet** Explorer may be used in at least one embodiment of the present invention. Although the present invention is described with respect to employing Java, it will be appreciated that any suitable programming language may be employed to carry out the present invention.

Detailed Description Text - DETX (45):

The client 110 includes item classes (e.g., DNA sequence item) 150 which represent views of data that reside on the server 112. For example a DNA sequence item that resides on the client 110 (Java side) is a **view manager** that knows how to display a DNA sequence. The DNA sequence item does not possess the physical data of the particular DNA sequence. Rather the DNA sequence item contains a set of methods that provide for rendering the DNA sequence on the screen 58 (FIG. 1) from data (on the server side located at the datastore) which represents the DNA sequence. Thus, the DNA sequence item provides for drawing the DNA sequence on the screen 58 by manipulating data that resides on the server 112. The client 110 simply stores a cache copy of the DNA data which is temporary.

Detailed Description Text - DETX (46):

If there is ever a change at the server end with respect to a persistent copy of data of a particular item that is currently being viewed by a client 110, the server 100 sends a callback to the client 110 informing the client 110 that it is viewing an obsolete version of the data. The server 112 then instructs the client 110 to refresh its contents with respect to that data item and to rerender it on the screen 58. Thus the client 110 serves as a **view manager** of persistent data residing on the server 112. The client 110 is notified when state changes occur, and the client 110 in response refreshes its cached memory to reflect the state change.

Detailed Description Text - DETX (58):

FIG. 8 is a drawing depicting a network architecture 300 according to one aspect of the present invention. It will be appreciated that various architectures may be employed to carry out the present invention, all which fall within the scope of the present invention. The architecture 300 consists of several network layers. The architecture 300 is divided up to show the architecture on the client side 310 and the architecture on the server side 320. The bottom layer 326 is the transport protocol layer. In the preferred embodiment, the transport protocol layer 326 is a **TCP/IP** layer--of course any suitable type of transport protocol layer may be employed to carry out the present invention. The transport protocol layer 326 exists on both the client side 310 and the server side 320. This layer 326 in general serves to transport data without error or loss.

Detailed Description Text - DETX (59):

On top of the transport layer 326 is a marshaling protocol layer 330 on both

the client side 310 and server side 320. In this embodiment, the marshaling protocol layer 310 is a IIOP ORB (**Internet** Interoperation Protocol Object Request Broker). The marshaling protocol layer 320 provides for connecting client objects with server objects. On the client side 310, over the marshaling protocol layer 330 is a client stub 332 which provides for gluing the project client 336 to the marshaling protocol layer 330, and it also provides for connecting the project client 336 to the server stub 340. The project client 336 serves to manage projects on the client side 310 for a particular client 110. The project client's 336 preferred implementation is Java, however, it should be appreciated that other programming languages could be used for the proj. client 336. In general, the proj. client 336 is a cached copy of a corresponding team project stored on the server 112. On top of the proj. client 336 is a client view layer 350 which provides for viewing data and analysis in general.

Detailed Description Text - DETX (60):

The CORBA 332 stub (written in Java) and the **view manager** 350 are downloaded at run time to create the client object class which is used to instantiate new objects in the server 112.

Detailed Description Text - DETX (75):

2. Transfer data to **Internet** programs